Constraint-based Self-adaptation of Wireless Sensor Networks

Nadia Gamez
Dpto de Lenguajes y Ciencias de la Comunicación
Universidad de Málaga
nadia@lcc.uma.es

Daniel Romero
ADAM Project-team
Université Lille 1, LIFL CNRS UMR 8022
INRIA Lille-Nord Europe, France
daniel.romero@inria.fr

Lidia Fuentes
Dpto de Lenguajes y Ciencias de la Comunicación
Universidad de Málaga
lff@lcc.uma.es

Romain Rouvoy
ADAM Project-team
Université Lille 1, LIFL CNRS UMR 8022
INRIA Lille-Nord Europe, France
romain.rouvoy@inria.fr

Laurence Duchien
ADAM Project-team
Université Lille 1, LIFL CNRS UMR 8022
INRIA Lille-Nord Europe, France
laurence.duchien@inria.fr

ABSTRACT

In recent years, the *Wireless Sensor Networks* (WSNs) have become a useful mechanism to monitor physical phenomena in environments. The sensors that make part of these long-lived networks have to be reconfigured according to context changes in order to preserve the operation of the network. Such reconfigurations require to consider the distributed nature of the sensor nodes as well as their resource scarceness. Therefore, self-adaptations for WSNs have special requirements comparing with traditional information systems. In particular, the reconfiguration of the WSN requires a trade-off between critical dimensions for this kind of networks and devices, such as resource consumption or reconfiguration cost. Thus, in this paper, we propose to exploit *Constraint-Satisfaction Problem* (CSP) techniques in order to find a suitable configuration for self-adapting WSNs, modelled using a *Dynamic Software Product Line* (DSPL), when the context changes. We exploit CSP modeling to find a compromise between contradictory dimensions. To illustrate our approach, we use an *Intelligent Transportation System* scenario. This case study enables us to show the advantages of obtaining suitable and optimized configurations for self-adapting WSNs.

WAS4FI'12 Bertinoro, Italy

Categories and Subject Descriptors

H.1 [**Models and Principles**]: Miscellaneous; D.2.10 [**Software Engineering**]: Design - methodologies

General Terms

Design, Performance

Keywords

Constraint-Satisfaction Problem, Dynamic Software Product Lines, Self-adaptation, Wireless Sensor Networks

1. INTRODUCTION

Wireless Sensors Networks (WSNs) refers to the set of sensor nodes connected by a wireless medium that are able to perform distributed sensing and convey useful information to control stations [2]. They are attracting huge interest due to their potential of applicability in a variety of pervasive systems, such as smart spaces, intelligent transportation systems or ambient assisted living applications. However, they impose several requirements, compared to traditional information systems, mainly due to their resources scarceness (*e.g.*, battery or memory).

An important issue in WSNs is their reconfiguration regarding changing conditions or context changes to reduce network degradations or to improve the functioning. Such reconfigurations have to be done considering sensor deployment in remote and unattended areas as well as the satisfaction of conflicting objectives (*e.g.*, accuracy versus energy consumption). The former requires WSNs having an autonomous behavior, *i.e.*, networks that exhibits "self-*" properties [9] in order to react by themselves to the context changes. The latter requires a decision making mechanism in the reconfiguration process that takes into account not only the context changes but also the resource scarceness of WSNs or other sensors specific characteristics to deal with conflicting objectives.

In order to achieve self-adaptations in WSNs, in a previous work [8] we proposed FAMIWARE, a family of middleware for Ambient Intelligence (AmI) systems, deployed in sensors devices and smartphones. FAMIWARE uses feature models and *Dynamic Software Product Lines* (DSPLs) to drive the self-adaptation process. The reconfiguration consists in replacing the current feature model configuration by a new configuration more suitable to the new context situation.

In this paper, we focus on the decision making mechanism of the reconfiguration process defined by FAMIWARE considering that multiple target configurations can satisfy the new context. Given the resource scarceness in WSNs, the resulting target configuration have to be optimal as possible regarding dimensions such as resource consumption and reconfiguration cost. However, such selection can not be done to the detriment of other dimensions such as the Quality of Service (QoS) offered by the network. Therefore, we propose a decision making mechanism for FAMIWARE, which deals with the trade-off of different dimensions by using multi-objective optimizations. Our approach provide the flexibility to specify the dimensions to consider for the self-adaptation. In this way, the resulting configuration respect the context but at the same time is the most suitable regarding additional critical dimensions in WSNs.

After this introduction, this paper is structured as follows. We start by describing a motivation scenario and identifying the challenges associated with the selection of the new network configuration (cf. Section 2). Then, we present FAMIWARE (cf. Section 3) the middleware that we extend in order to improve the self-daptation process (cf. Section 4). We continuous with the discussion about the advantages of our work (cf. Section 5) before discussing some related works (cf. Section 6). We finish with some conclusions and perspectives of our work (cf. Section 7).

2. MOTIVATION & CHALLENGES

In order to motivate our work, we use an *Intelligent Transportation System* (ITS)[1] scenario. This kind of systems uses sensors, on board computers, GPS and other devices to improve the mobility, safety, and security, while ensuring energy efficiency and reducing environmental impacts of transportation systems. In a particular ITS application, we have several static sensors placed in a road sensing movement, light, noise and temperature in order to collect data about traffic, possible accidents or environmental issues. In the feature model of Figure 1, a WSN is made up of one or more sensors (feature with 1..* cardinality). An important characteristic of this kind of networks is its routing protocol that allows the communication between sensor nodes. In this feature model are represented six routing protocols: DD, Drip, ACM, TinyHop, AODV and TYMO. The protocol running in every node has to be the same for all the sensors (cf. *RProtocol* feature and its xor children in Figure 1), but also the sensors can have optionally preinstalled other protocols (cf. *Routing* feature in Figure 1). In addition, other characteristics of sensors relevant for our scenario are the role that they can play in the network (ordinary node, cluster-head, or sink), the state of the node (alive or slept) and the frequency of the sensing tasks. Let us suppose that the road in the scenario becomes a secondary road because a nearby highway is open. Then, the use of the road is drastically reduced and the information collected by its sensors is not as critical as before. To increase the lifetime of the network, the system can be reconfigured to save energy. To do that, several options are possible:

- *Change of the routing protocol*: Figure 2 depicts the feature diagram of one partial configuration of the network with some energy efficient protocols, *i.e.*, TYMO, TinyHop and AODV. To save energy in this configuration, we can select the most energy efficient routing protocol namely AODV. However, by doing that we could impact the response time of the whole network since this protocol is not necessarily the fastest protocol. Additionally, if we choose the AODV protocol which is not preinstalled in several nodes, we produce additional reconfiguration costs in terms of energy expense by sending large size messages that contain the protocol functionality.
- *Deactivation of sensor nodes*: Another way to save energy is by sleeping nodes on the network. For example, we can decide to deactivate the 50% of nodes. But, how do we select the candidates? We could do it randomly or arbitrarily. However, this deactivation has to be executed carefully because we can sleep a cluster head node or a sink making the network or part of it useless. In a similar way, depending on the network topology, we can sleep several consecutive nodes seriously affecting the network accuracy.
- *Reduction of the monitoring frequency*: Finally, we can reduce the frequency of the monitoring tasks in sensors. Nevertheless, if the WSN response time increases a lot, we cannot ensure more the safety of the transportation system.

By using these different alternatives, we can find several valid configurations that will reduce the energy consumption in the network. However, as we just mentioned, we cannot only consider the energy saving to select the new configuration. We need also to include other aspects of the network such as reconfiguration cost and QoS properties (*e.g.*, accuracy), which lead us to find contradictory objectives.

Challenges.

From the previous scenario we have identified the following challenges:

1. **How to select the best fitting target configuration:** As already said, WSNs are characterized by resource scarceness. The message exchange as well operations required to reach the new configuration (*e.g.*, code installation and changes on nodes properties) have to be minimized in order to preserve the network resources and the correct operation. Therefore, the new selected configuration of the WSN should be the most adequate one, regarding specific dimensions such as energy consumption or accuracy of the sensed information. Thus, one of the challenge will be to identify the dimensions to optimize the targets configurations.
2. **How to deal with conflicting objectives:** In the process of finding the most adequate target WSN

[1]IEEE Intelligent Transportation Systems Society: http://sites.ieee.org/itss/

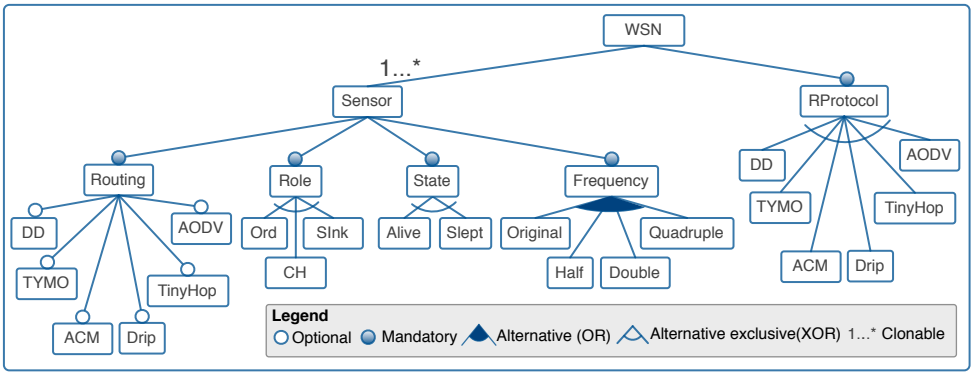

Figure 1: Feature Model for the ITS Scenario

configuration for a specific context could focus in one dimension, sacrificing others. For instance, installing a new energy efficient routing protocol in all the network could be very expensive in terms of reconfiguration cost. In this case, the energy saving and the reconfiguration cost dimension can be considered as contradictory objectives. Then, in order to solve this kind of situations, we will need to consider multiple criteria in the selection process. It is necessary to find a suitable trade-off between such objectives.

In the following section we present the FamiWare middleware, which we extend in order to deal with the mentioned challenges.

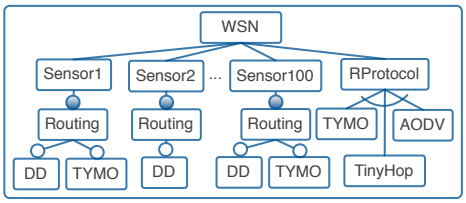

Figure 2: Network Configuration with Energy Efficient Protocols

3. SELF-ADAPTATION IN FAMIWARE

FamiWare is a family of middleware for *Ambient Intelligence* (AmI) systems to be deploy in several kind of devices as sensors. FamiWare applies the *Software Product Line* (SPL) [11] approach to characterize the inherent variability of the AmI domain by means of *Feature Models* (FM) [10]. This FM is the base of a model driven process that derives a configuration adapted to the requirements of each system device. Furthermore, FamiWare has a mechanism to achieve self-adaptation by applying the Autonomic Computing paradigm at middleware level [8].

3.1 FamiWare Reconfiguration

As part of the autonomic computing architecture developed in FamiWare, in every instance of the middleware there are several monitoring services observing the context. The sensed data are analyzed by the context-aware service to check on context changes. When a context change is detected, the self-adaptation process is triggered.

Figure 3 summarizes the process to configure and to reconfigure FamiWare. At design time, the *middleware architect* defines the feature model that represents the family of middleware ①. Then, this family is customized by following the application requirements provided by the *application developer* ②.

As we mentioned before, in FamiWare the reconfiguration is driven by feature model runtime configurations. Then, the first step to reconfigure the system is to find the target runtime feature model configuration fitting the new context. The Hydra feature modeling tool[2], used by FamiWare, is able to find a set of valid configurations (called feature model specialization) respecting the new context ④. However, we need to select the most suitable configuration in this set considering additional information. Therefore, as contribution of this paper, we extend FamiWare by allowing *application developer* to specify such dimensions ③, which are used by a CSP service to find the new configuration ⑤.

Finally, once the target configuration is obtained, the reconfiguration service provided by FamiWare is responsible for executing the required actions to reach that target configuration. But the reconfiguration execution is out of the scope of this paper. Here we focus only in the decision of the adequate target configuration.

3.2 Reconfigurable Parts in FamiWare

In order to identify what dimensions we can optimize in the self-adaptation process, firstly we have to specify the parts of FamiWare we can reconfigure.

The internal architecture of FamiWare is composed by a microkernel and several provided services. The microkernel is responsible for the composition of the application and the services. The microkernel also exposes a data delivery service to transmit the data over the network using a specific network routing protocol. Every particular middleware configuration is installed in each sensor node of the network. Then, we can reconfigure hardware of every sensor node or its internal FamiWare architecture. Regarding hardware

[2]The Hydra feature modeling tool: `http://caosd.lcc.uma.es/spl/hydra`

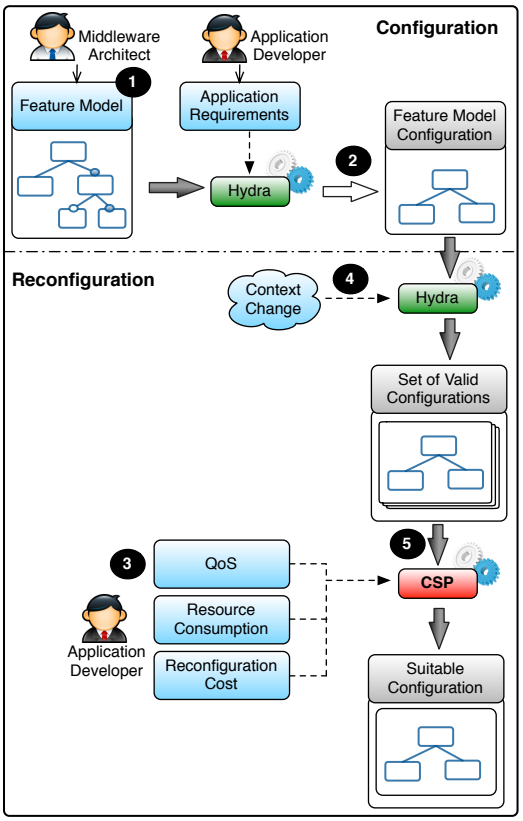

Figure 3: FamiWare Configuration and Reconfiguration Process

reconfigurations, we can switch off and on the radio chip of the sensor or change the whole state of the sensor (*e.g.*, slept, idle or alive). The reconfigurations in the FamiWare software architecture include coarse-grained modifications, like removing or adding a service or changing the routing protocol, and fine-grained modifications, like changing some internal parameters of a service.

In the next section, we introduce our proposal to improve the selection of the new configuration of the network in FamiWare.

4. PROPOSAL

In FamiWare, the reconfiguration process is based on the context information and can have different granularity levels (from parametrization to component replacement). In Section 3.1, we have provided an overview of the whole reconfiguration process. In this section, we focus on how to find the most suitable feature model configuration to adapt the system to a new context situation.

We propose that in order to improve the functioning of the WSNs, a new configuration has not only to satisfy the context but also to improve the network regarding additional aspects. For example, in our motivation scenario (cf. Section 2), we search configurations that reduce the energy consumption. But, in this search we need also to consider dimensions such as accuracy or response time, which are relevant taking into account the network objectives. A network that is energy efficient but has a low accuracy or takes a lot of time to provide information is not acceptable in the case of intelligent system transportations. Thus, the consideration of different dimensions can lead to conflicting objectives.

To deal with these conflicts, in this section we define an approach based on Constraint Satisfaction Problem (CSP) methods, to find a suitable configuration of the WSN considering multiple dimensions such as resource consumption and Quality of Service (QoS) at the same time. The advantage of our approach is double: *i)* to provide a flexible reconfiguration process that allows application developers to refine the selection of the new network configuration when required and, *ii)* to deal with conflicting objectives. We start with the identification of the dimensions that we can use to select the new configuration by exploiting FamiWare and then we present the CSP modeling of the configuration selection.

4.1 What dimensions we can measure?

In order to optimize the self-adaptation of the WSNs, we need to identify dimensions that will be used in the selection process. To do this, we consider the quality of the network operations, the impact related to the reconfigurations actions and, of course, the maximization of the network life as relevant for the reconfigurations of WSNs. They are reified by the following dimensions:

- *Quality of Service (QoS)*: In WSNs, QoS properties such as latency and accuracy of the sensed information may be critical. The latency to retrieve the monitored data from each node depends on the frequency of the sensing tasks but also on the selected routing protocol. There are routing protocols that send many messages to ensure that the information arrives to the destination as soon as possible, but this has a penalty in the energy saving, since communications have a high cost. On the other hand, the accuracy of the sensed information will depend on the number and position of active nodes that are sensing this data. In our scenario (cf. Section 2), to save energy in the whole network we can deactivate some nodes during a period of time and activate them again when the battery of rest of the nodes drops. In this way, we extend the life of the global network, but of course, the accuracy is reduced.

- *Reconfiguration Cost*: It is related to the number and size of exchanged messages for realizing the reconfiguration. As stated by [1], the communication tasks consume more energy that the processing task in tiny devices such as sensor nodes. Therefore, we prefer parameterization on the installation of new functionality when possible, since this implies to send large messages with pieces of code. For example, if we need to choose between the addition of a new component or the use of a preinstalled component in a sensor node, we give the priority to the second choice. Thus, we reduce the size of the exchanged messages.

- *Resource Consumption*: The last dimension considered in the optimization of the reconfiguration is the resource consumption of the target configuration. In WSNs, we need to save resources in order to maximize the operation time of the network. In particular, we can reduce the energy consumption by reducing

Table 1: Objective functions.

RCReCos	$\min\ (\sum_{i=1}^{i=\lvert\mathcal{S}\rvert} (1 - I_i(P_j)) + E(P_j)),\ P_j \in \mathcal{P}$
QoSRC	$\min\ (\sum_{i=1}^{i=\lvert\mathcal{S}\rvert} a(SN_i))$
ReCosQoS	$\min\ (\sum_{i=1}^{i=\lvert\mathcal{S}\rvert} sel(SN_i) * Y_i)$

communications, the frequency of monitoring tasks, but also disabling sensor nodes when they are not required.

The previous dimensions are computed from information kept or employed by FamiWare such as node state and frequency of sensing tasks. This means that their usage in the reconfiguration process does not introduce an additional overhead in terms of energy consumption. In the following section, we present our model applying CSP in order to include different and conflicting objectives based on the presented dimensions.

4.2 CSP Model in FamiWare

As we have already said, when we reconfigure the wireless sensor network considering several dimensions we can find conflicting objectives. For example, the reduction of the energy consumption by deactivating nodes impacts the accuracy of the sensed information. In a similar way, the usage of a routing protocol that consumes a low quantity of energy can have a high reconfiguration cost if such protocol is not deployed on the whole network. Our model deals with these conflicts by means of CSP. In particular, we improve the decision-making service from FamiWare with such a model (cf. Figure 3). Once, FamiWare has identified a set of target configurations dealing with the new context, our approach gives to application developers the opportunity of defining dimensions which constitute an additional filter to reach the new target configuration. In this way, at the end of the reconfiguration process, we will have a network configuration that is suitable considering multiple dimensions. Although our approach is based on FamiWare capabilities, it can be extended and generalized to include other dimensions or other generic WSN applications.

In this section, we explore the different optimizations based on several combinations of the mentioned dimensions.

4.2.1 Optimization based on the Resource Consumption and the Reconfiguration Cost

In this kind of optimization we can deal, for instance, with the *trade-off* between energy consumption and the reconfiguration cost. We search to choose a new configuration with an energy efficient routing protocol. However, at the same time, as communications are expensive in WSNs, we also need to reduce the reconfiguration cost by using the protocol that, if it is possible, was previously deployed on most sensor nodes. The *RCReCos* function in Table 1 searches to satisfy both objectives by minimizing the numbers of nodes where the new protocol must be installed and the energy used by the protocol.

The expression $I_i(P_j)$ returns 1 if the i^{th} sensor node has pre-installed the P_j protocol (which is part of the set $\mathcal{P}$ of protocols used in the network) or 0 on the contrary case. $E(P_j)$ retrieves the energy used by the P_j protocol, which value is extracted from the protocol specification and experimental results.

Below we present the only constraint required in the *RCReCos* optimization:

C1$_{RCReCos}$ ($\forall\ P_i, P_j | P_i \in \mathcal{P} \wedge P_j \in \mathcal{P} : (s(P_i) = 1 \wedge s(P_j) = 1) \Rightarrow P_i = P_j$): *Only one protocol is chosen for the new configuration.*

In our ITS scenario we have six possible routing protocols, as it was shown in Figure 1, but not all of them are energy efficient. In fact, only three (TYMO, TinyHop and AODV) are the ones that FamiWare selects as suitable protocols for saving energy as it was shown in Figure 2. FamiWare obtains the set of valid configurations of the figure, taking as input a previous running feature model configuration and the new context constraints (*e.g.* new enery saving situation). In this figure, we can see that the DD and TYMO protocols were the protocols deployed on the different sensor nodes and that DD, DRIP and ACM protocols have been removed of these configurations since they are not energy efficient. This classification of protocol is made by FamiWare basing in previous simulations. In order to define the new configuration, we apply *RCReCos* on TYMO, TinyHop and AODV. AODV is the more energy efficient protocol over the other two but TYMO protocol has the lowest value for *RCReCos*. It is expected as TYMO is deployed on most of nodes and the cost for reconfiguring the system with AODV that is is not pre-installed is higher.

4.2.2 Optimization based on Quality of Service and Resource Consumption

Here, we search for a balance between the accuracy offered by the whole network (QoS) and the consumed energy (resource consumption). The *QoSRC* objective function models this optimization. In this function we minimize the number of active sensor nodes in the network. In particular, the expression $a(SN_i)$ indicates if the SN_i node is active (1) or not (0).

Since sink and cluster-heads manage nodes, their deactivation introduce an additional cost on the reconfiguration because other nodes would have to take their responsibilities. Then, in our model, sink and cluster-heads cannot be deactivated. Therefore, we have the following basic constraints in order to optimize *QoSRC*:

C1$_{QoSRC}$ ($\forall\ SN_i | SN_i \in \mathcal{CH} : a(SN_i) = 1$): *All the nodes that are cluster head are always active.*

C2$_{QoSRC}$($\forall\ SN_i | SN_i \in \mathcal{SIN} : a(SN_i) = 1$): *All the sink nodes are active.*

With the previous constraints, the minimization of active nodes will get a configuration where all the ordinary nodes will be deactivated, but this would be inadmissible. In a similar way, in the ITS, it does not have sense to sleep all the first sensors placed in the road and only maintain active the last ones. Then, we have to avoid the deactivation of many consecutive nodes in order to get a suitable accuracy. For this reason, we define the following constraint:

C3$_{QoSRC}$ $1 - la_N \geq X,\ X > 0$: *The network offers at least an accuracy of X.*

In this constraint, we assume that the accuracy is between 0 (excluded) and 1 included. The la_N value represents the lost accuracy in the network in function of the nodes that are deactivated. To calculate la_N we use the algorithm 1.

In this algorithm, we assume that all the nodes have the same contribution on the accuracy. Therefore, we compute the lost accuracy as the arithmetic mean of the deactivated nodes (cf. line 5, algorithm 1). However, in WSNs the accuracy is also impacted by the distance between nodes that are still active. Then, in line 7 we look on the neighbors of each node and increase by m^{-2} the lost accuracy when one of the neighbors is not active, with m being the number of nodes in the network. As one of the preconditions of the algorithm is that sink and cluster heads are always active (*i.e.*, $a(SN) = 1$ for all SN sink or cluster head), the la_N never will be greater than one. Furthermore, the satisfaction of this condition is guarantee by $C1_{QoSRC}$ and $C2_{QoSRC}$ constraints.

Algorithm 1 Computation of lost accuracy la_N value

Require: A set of sensor nodes $\mathcal{S}$
Require: A set of neighbors $\mathcal{N}_{SN}$ for each $SN \in \mathcal{S}$
Require: $(\exists SN | SN \in \mathcal{S} : a(SN) = 1)$
Ensure: The lost accuracy la_N according to the deactivated sensor nodes

```
1:  la_N ← 0
2:  m ← S.size()
3:  for all (SN ∈ S) do
4:    if (SN.activate() = false) then
5:      la_N ← la_N + (1/m)
6:      SN.visit(true)
7:      for all (NE ∈ N_SN) do
8:        if (NE.visited() = false and NE.active() = false)
          then
9:          la_N ← la_N + (1/m^2)
10:         NE.visit(true)
11:       end if
12:     end for
13:   end if
14: end for
```

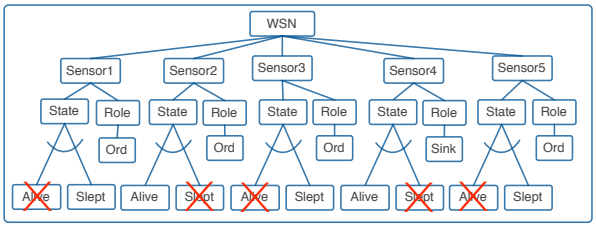

Figure 4: Network Configuration with a Minimal Accuracy $X = 0.6$

Figure 4 depicts a simplification of the ITS scenario with five consecutive nodes (*i.e.*, $m = 5$). The *Sensor4* is a sink, therefore it will be always active. Considering $X = 0.6$, we can not just leave *Sensor4* active because la_N would be 0.92. With *Sensor1* and *Sensor4* active, $la_N = 0.68$ and with *Sensor3* and *Sensor4* active, $la_N = 0.64$. If 1, 3 and 5 are active, $la_N = 0.4$ and 1, 2, 3 and 5 are active, $la_N = 0.2$. However, with the *QoSRC* objective function we minimize the number of active nodes, and then we prefer the solution with three active nodes that respects $C3_{QoSRC}$. Therefore, nodes 2 and 4 are deactivated and the other nodes are kept active as depicted by Figure 4.

4.2.3 *Optimization based on the Reconfiguration Cost and the Quality of Service*

With this optimization, in order to save energy, we want to reduce the monitoring frequency (QoS) of the different nodes in the network. But at the same time, we search to reduce the reconfiguration cost, which means to select a minimal number of nodes to reconfigure. To do that, we define *ReCosQoS*. In this function, we use $sel(SN_i)$ expression which indicates if the node has to be reconfigured. Y_i represents the factor to reduce the monitoring frequency of the i^{th} sensor node.

From the *ReCosQoS* definition, it is clear that we want to minimize the number of nodes to reconfigure. However, to guarantee a suitable monitoring frequency (QoS) we include the following constraint in the optimization:

$\mathbf{C1}_{ReCosQoS}$ $(\forall\ SN_i | SN_i \in \mathcal{S} : ((mf(SN_i) * Y_i) \geq X_{1i}) \wedge ((mf(SN_i) * Y_i) \leq X_{2i}) \wedge (X_{1i} \leq X_{2i}) \wedge (X_{1i} \geq 0))$: *All the sensor nodes have a monitoring frequency in a given interval.*

In this constraint, the $mf(SN_i)$ indicates the current monitoring frequency of the SN_i node. On the other hand, X_{1i} and X_{2i} are the lowest and highest frequency that the SN_i node can have in the new configuration of the WSN. Such values are specified by the application developer. The minimal Y_i factor for the i^{th} node is determined according to $mf(SN_i)$ and the $C1_{ReCosQoS}$ constraint.

5. DISCUSSION

5.1 Implementation Considerations

In order to validate our approach, we have implemented a prototype of CSP service that includes the objective functions described in Section 4. We propose a modular architecture of the service to make the addition of new optimizations easier and foster their reuse. In this way, the three optimization described in Section 4 are implemented in three different modules.

To reduce the overhead introduced by the optimizations at the execution, we avoid the usage of a CSP solver when possible. Additionally, we benefit from the data holding by FamiWare (cf. Section 4.1) to compute some information in advance. In particular, for the optimization based on the resource consumption and reconfiguration cost (cf. Section 4.2.1) we are able to compute the value of the *RCReCos* function for each protocol. We keep a list sorted in ascending order of those values. In this way, we select just the protocol which value is on the top of the list. The *RCReCos* values are updated just when we decide the installation of protocols during the reconfiguration. In those cases we use an efficient sort algorithm such as quicksort, to introduce the new value into the list.

Regarding the optimization related to accuracy and energy saving (cf. Section 4.2.2), we reduce the overhead by generating, at design time, a table containing the $1 - la_N$ values for the different configurations of active and deactivated nodes. For each row in the table, we have the $1 - la_N$ value and the list of nodes to be deactivated to reach such value. The rows are ordered in descending order. This ordering allows us to apply a binary search algorithm to find the configuration that satisfy the X value for the accuracy.

Table 2: CSP results for the ReCosQoS optimization.

Test	Feature Number	Sensor Number	CSP Latency(ms)
a)	57	4	0.08
b)	1305	100	1.35
c)	6505	500	6.89
d)	13005	1000	25.05
e)	52005	4000	362.73

The number of rows in the table is $\sum_{k=1}^{k=m} \binom{m}{k} = 2^m - 1$, where m represents the number of ordinary sensor nodes. In the table generation, we apply the algorithm introduced in Section 4.2.2, which have a complexity of $O(n^2)$ in the worst case, when the WSN topology is fully connected.

In the case of the *ReCosQoS* function optimization we used the JaCoP (Java Constraint Programming) solver[3]. The selection of this library is motivated because of its simplicity and spread usage in the scientist community. The X_{1i} and X_{2i} values are defined by the developer at design time. We measure the overhead of the optimization by executing several tests with different configurations. Each test was executed 10000 times. We calculated the average time of the execution by excluding the first 1000, which were considered as part of the warm-up. Table 2 summarizes the obtained results. In the different tests, we use the energy efficient configuration depicted in Figure 2 which has, for each node, 13 features to model the routing protocols, state, role and frequency of the node. In each test we varied the number of sensors nodes in the network. As observed, for networks with less of 1000 sensor nodes, we rest under $30ms$ (cf. tests a to d). With more complex networks configurations (cf. test e) the latency increases considerably even if it remains less than one second. However, the different tests confirm that we can use a CSP solver in the optimizations when required with a reasonable overhead.

5.2 Benefits of the Approach

In FamiWare the selection of a target configuration takes less than a minute for a feature model configuration with 4000 features. With our approach we introduce an additional overhead of $7ms$ for 6500 features (cf. Table 2, configuration c). With a more reasonable feature model configuration of 100 features, FamiWare is able to find the valid configurations in one second while our approach increases this time in $1ms$ for 60 features. Therefore, as expected, the usage of CSP in the reconfiguration process introduces an additional cost. However, this cost is negligible considering the benefits of our approach, which we detail now.

First, the usage of additional dimensions derived from the same context information enables FamiWare to find a configuration that not only respect context but also optimize the reconfiguration itself. For instance, in our ITS scenario without using our proposal, FamiWare chooses TinyHop (the more on the left subfeature of *RProtocol* in Figure 2) to save energy changing the routing protocol. In our scenario, this protocol is not preinstalled in several nodes. Therefore, the reconfiguration cost is higher since the code with the new protocol has to be spread through the network. This would implies a spend of energy, since the most costly operation is the communication. In our approach, we actually seek a better use of the reconfiguration opportunities regarding the resource scarceness in WSNs.

Second, our approach searches to improve the network configuration regarding multiple and conflicting objectives. In general, it is not suitable to configure the network by considering only one objective since other aspects can be affected. Continuing with our ITS scenario, we cannot just reduce the energy consumption by deactivating nodes. For example, FamiWare does not consider the penalty in the accuracy when consecutive sensor nodes are not active. This means that the configuration chosen by the middleware platform could sleep several consecutive nodes having as a result a network that will not provide right information about the entire road. With our CSP based solution, we find a compromise between this kind of conflicts.

Third, the integration of our approach with FamiWare is easily configurable to be used just when it is required. In the case of our scenario, the execution time of the whole reconfiguration triggered by context changes is not critical. Objectives like the reduction of the energy consumption or keeping a degree of accuracy are more important. Then, we can applied our CSP based solution. Nevertheless, in cases where the time in the reconfiguration is critical such as emergency situations, the optimizations can be ignored. In these cases, it is only important reconfigure the network to deal with the new context. Then, we will choose one of the next target configuration for the new situation without regarding other dimensions. This new configuration will be not necessarily the most suitable to extend the WSN live but at least it will works according to the new context.

Finally, in our approach we do not consider the combinatorial explosion related to variants. In fact, it worths notice that FamiWare defines a high amount of dependencies between the features in its feature model. This reduces the combinatorial explosion of possible variants. Furthermore, as what we propose here is to reason about the set of possible configurations provided by FamiWare, it is out of the scope of this paper to discuses about the combinatorial explosion problem.

6. RELATED WORK

In the literature, we find several works [14], [3], [7], [4], [5] dealing with the adaptation of WSNs with similar motivation as our approach. However, many of them [14], [3], [7] focus in how to realize this adaptation over the network instead to search for a suitable configuration of the whole system as we propose. In [14], an adaptation mechanism based in an algorithm to detect coverage and topology of sensors is presented. They use the life time of the network as a goal as we do. Similarly, in [3], the energy is the most important requirement in the adaptation of the WSNs. However, again they do not explain which will be the reconfiguration of the system. They focus on how to reconfigure using a mechanism of code injection. Authors in [7] present a reflecting middleware for WSNs that adapts the network to maintain the QoS requirement. Nevertheless, they only pay attention to QoS dimension instead to provide a mechanism to optimize the network regarding the context changes and other

[3]JaCoP Solver: `http://www.jacop.eu/`

dimensions. Finally, in [4] and [5] the authors try to deal with the adaptation taking into account conflicting objectives. However, they use a biological adaptation mechanism to reconfigure the system regarding only the latency, cost and success rate. Instead, our dimensions are more generic and cover more objectives.

Out of the WSN domain, CSP, Dynamic Software Product Lines and Feature Models are widely used to perform self-adaptation. [13] uses feature models and a heuristic algorithm to derive configurations that meet resource constraints. However, these constraints are more relaxed that the constraints that we must deal with for the sensors devices. In [6], product lines are applied to support self-adaptative applications. In this work, the authors focus in dealing with the the combinatorial explosion of variants. As we mentioned in the past section, FamiWare reduce this problem exploiting the dependencies between features. Furthermore, in this work we only focus in the selection between a bounded set of configurations. In a similar way, in [12] tailor Dynamic Software Product Lines feature models. They try to bridge the gap between the features and the component-based runtime adaptation. Then, they go a step forward than us, since our purpose is to choose the new feature model configuration and then to map this configuration into the architecture of the system.

7. CONCLUSIONS

In this paper we presented an approach to enable the selection of a new WSN configuration when multiple configurations satisfy the current context. In particular, we applied multiobjective optimizations to deal with conflicting objectives. The optimizations are based on the resource consumption, the QoS offered by the network and the reconfiguration cost, which are important dimensions to consider in the configuration to be reached. Our approach is integrated into the FamiWare middleware that provides a feature driven reconfiguration support for WSNs.

Given the limited capabilities of WSNs, our approach searches to improve the reconfiguration process looking for a suitable configuration of the network. Our discussion shows that in some situations we can avoid the usage of CSP solvers to reduce the overhead introduced by the optimizations. Nevertheless, we can still use solvers with a reasonable overhead as it is confirmed by the executed tests. Additionally, as the reconfiguration time of the network can be critical in some situations, we also provide the flexibility to use it only when considered appropiate.

Future work includes the definition of new objective functions considering other dimensions such as reliability and data routing. We also plan to extend our optimizations to enable software modifications in the sensor nodes. Currently, we are limited to parametrization of the network. By benefiting from the FamiWare capabilities in terms coarse-grained modifications like removing or adding services in the nodes we can also optimize the software running on the sensor nodes.

8. ACKNOWLEDGMENTS

This work has been partially supported by the projects TIN2008-01942 funded by Spanish Ministry of Science and Innovation and P09-TIC-05231 (FamiWare) funded by Andalusian Government.

9. REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, aug 2002.

[2] I. F. Akyildiz and I. H. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks*, 2(4):351 – 367, 2004.

[3] M. Avvenuti, P. Corsini, P. Masci, and A. Vecchio. An application adaptation layer for wireless sensor networks. *Pervasive Mob. Comput.*, 3:413–438, August 2007.

[4] P. Boonma and J. Suzuki. Evolutionary constraint-based multiobjective adaptation for self-organizing wireless sensor networks. In *Bio-Inspired Models of Network, Information and Computing Systems, 2007. Bionetics 2007. 2nd*, pages 111 –119, dec. 2007.

[5] P. Boonma and J. Suzuki. Monsoon: A coevolutionary multiobjective adaptation framework for dynamic wireless sensor networks. In *HICSS*, page 497, 2008.

[6] G. Brataas, S. O. Hallsteinsen, R. Rouvoy, and F. Eliassen. Scalability of decision models for dynamic product lines. In *Software Product Lines, 11th International Conference, SPLC 2007, Kyoto, Japan, September 10-14, 2007, Proceedings. Second Volume (Workshops)*, pages 23–32. Kindai Kagaku Sha Co. Ltd., Tokyo, Japan, 2007.

[7] F. C. Delicato, P. F. Pires, L. Rust, L. Pirmez, and J. F. de Rezende. Reflective middleware for wireless sensor networks. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 1155–1159, New York, NY, USA, 2005. ACM.

[8] N. Gámez, L. Fuentes, and M. A. Aragüez. Autonomic computing driven by feature models and architecture in famiware. In *Proceedings of Software Architecture - 5th European Conference, ECSA 2011, Essen, Germany, September 13-16, 2011*, pages 164–179, 2011.

[9] Ibm. An architectural blueprint for autonomic computing. *Quality*, 36(June):34, 2006.

[10] K. Lee, K. C. Kang, and J. Lee. Concepts and guidelines of feature modeling for product line software engineering. In *Proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools*, ICSR-7, pages 62–77, London, UK, UK, 2002. Springer-Verlag.

[11] K. Pohl, G. Böckle, and F. J. v. d. Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[12] M. Rosenmüller, N. Siegmund, M. Pukall, and S. Apel. Tailoring dynamic software product lines. In *Proceedings of the 10th ACM international conference on Generative programming and component engineering*, GPCE '11, pages 3–12, New York, NY, USA, 2011. ACM.

[13] J. White, B. Dougherty, H. D. Strowd, and D. C. Schmidt. Software engineering for self-adaptive systems. chapter Using Filtered Cartesian Flattening and Microrebooting to Build Enterprise Applications with Self-adaptive Healing, pages 241–260.

Springer-Verlag, Berlin, Heidelberg, 2009.

[14] H. Zhang, P. Nixon, and S. Dobson. Multi criteria adaptation mechanisms in homological sensor networks. In *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on*, pages 937 –942, nov. 2008.